Putting it All Together: Software Planning, Estimating and Assessment for a Successful Project

Tim Kurtz
Science Applications
International Corporation,
NASA Glenn Research Center
21000 Brookpark Rd, MS 501-4
Cleveland, Ohio 44135, USA
Tim@bfsng.com
http://tkurtz.grc.nasa.gov/pete

Martin S. Feather
Jet Propulsion Laboratory,
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109, USA
Martin.S.Feather@Jpl.Nasa.Gov

Objective

How can project managers best incorporate estimation, planning, risk management, corporate processes and resources into a coordinated, tailored approach to developing and managing software? Large software efforts face the challenges of (1) identifying software risks, (2) judiciously planning for IV&V, development & QA activities to mitigate risks, (3) estimating cost & schedules of these plans, (4) accommodating the priorities of the primary stakeholders while ensuring mission success, (5) complying with standards & processes. This paper describes a coherent approach and accompanying tool support that addresses these challenges. The approach consists of three phases. The first phase characterizes the project, the second phase balances the risks in the project with the resources available to mitigate them and the last phase combines the results into plans for controlling the project. Two tools, Ask Pete, developed at NASA Glenn Research Center and DDP, developed at the NASA Jet Propulsion Lab, combine to assist project managers in completing the activities in these phases.

RESEARCH FUNDED BY NASA OSMA AND GSFC IV&V FACILITY

Characterize the Project

The first phase of the approach uses a user-friendly electronic interview to elicit overall project characteristics. Behind the scenes, relevant portions of this data are automatically fed into COCOMO II to yield cost and schedule estimates, and into other matrices to yield control level determinations and IV&V estimates. The COCOMO estimates, and other portions of the user-provided data, are then combined with institutional practices and policies (for example, ISO 9001, CMM and site-specific principles), yielding estimates of cost, schedule, and risk; and plans for the development effort, oversight and risk mitigation while conducting software development.

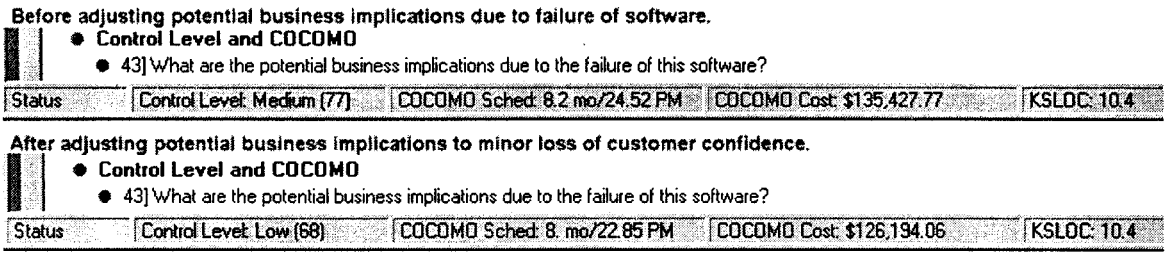

Figure 1, Change to Control Level Shown in Status Bar

Estimate Cost and Schedule

Determine the SLOC estimate using COCOMO II to scale the level of effort and develop an estimate of the cost and schedule based on what is known about the project, personnel, organization and resources. The COCOMO II questions form the foundation for this phase, providing much of the information needed for the control level and IV&V effort determinations. These initial schedule estimates form the basis for planning the project. The responses can be tuned, to a degree, to adjust the time and resources required to complete the project. Tradeoffs and the effects of software reuse can be examined by monitoring the Ask Pete status bar, see Figure 1.

Determine the Development Framework

Using the matrix in Table 1, identify the risks in three areas. These areas lay the foundation for identifying the level of control that must be implemented to minimize risk while optimizing the use of resources and maximizing the likelihood of successful project completion.

Identify Operational Risks

Identify the operational risks to the equipment, and personnel and the mission if the software should fail or be degraded. These risks can directly impact the chances for mission success.

Operational risks deal mainly with what will happen if the software fails during its functional use. Identify effects of the failure on personnel, the platform the software controls or operates as well as other equipment which may be negatively affected by adverse behavior and the ability for the mission to be successfully completed. As shown in the accompanying matrix, these risks can be large enough to require the imposition of Critical Control on the project irrespective of the magnitude of the other risks.

• Table 1, Control Level Matrix

Factor	1	2	3	4	5
Resourcing					
Software cost annualized over life of project (including all civil servants and contractors)	$100K - $500K	$500K – 1M	$1M - $2M	$2M - $20M	$20M and up
Organizational Complexity					
Project development location	Own Group	Several Groups, most at GRC	More than 2 other sites	More than 3 other sites	Numerous sites
Customers	Self	Other in own Directorate or one customer group, low number of users	Within GRC	Within NASA	Entire Industry or multiple industries
Developers Software experience level	All team qualified and experienced	Most team members qualified and experienced	Half the team qualified and experienced	Few team members experienced	Experienced staff not available
Technical Complexity					
Test Risk	No testing required	Minimum Testing	Standard testing required	Integrated Testing	Major testing effort required (e.g. IV&V)
Degree of Innovation	Well proven, known to GRC	Proven with some GRC experience	Proven, but new to GRC	Partially proven with some pioneering	Pioneering
Software development tool availability	All software development tools already purchased/in-house/familiar with	Majority of the software development tools are purchased /in-house/familiar with	Software development tools must be identified and purchased and learned	Majority of software development tools must be obtained, remainder developed	All software development tools must be developed
Interdependencies of deliverables	Simple standalone	Some Integration	Integrated	Highly integrated	Fully integrated
Safety Implications * (see NASA STD 8719.13A, NASA GB-1740.13 & NHB 1700.1 (V1-B))					
Potential Damage to carrier vehicle, major equipment, or system itself	No damage	Small/minor damage to equipment, or to system itself, mission still possible	Repairable/ recoverable damage to system and little or no damage to any related or surrounding systems	**Loss of system, and/ or damage to any critical surrounding systems or carrier vehicle**	**Loss of carrier vehicle (e.g. space craft, aircraft, major satellite)**
Potential Injury to personnel	No injury	Minor injury	Injury	**Severe injury or temporary disability**	**Loss of life or permanent disability**
Business Implications					
Consequence of Failure	Minor loss of Customer Confidence	Unsatisfied Customer	Damage to GRC Reputation	Damage to NASA reputation	**Significant impact to USA**
Schedule Pressure	No time pressure	Little effort to meet milestones	Nominal effort to meet milestones	Aggressive effort to meet milestones	Time critical

Critical Control

|-----Low control software -----|

|-- Medium control software--|

|---------- High control software ----------|

RESEARCH FUNDED BY NASA OSMA AND GSFC IV&V FACILITY

Identify Organizational Risks

Identify the risks to the development team and organization in successfully completing the project or if the software should fail or otherwise perform adversely.

Organization risks are concerned with the amount of resources committed to the project, the cohesiveness, quality and experience of the development team(s), the exposure the completed project will be subjected to, and the affect on the public's perception of the organization if the software or development effort should fail.

Identify Development Risks

Identify development risks that could affect successful completion of the project. Successful completion is defined as on time, within budget and with the proper functionality.

Development risks include the level of testing or IV&V required, the incorporation and development of new technologies, the availability and use of development tools and the amount of integration which must be achieved with other software or systems.

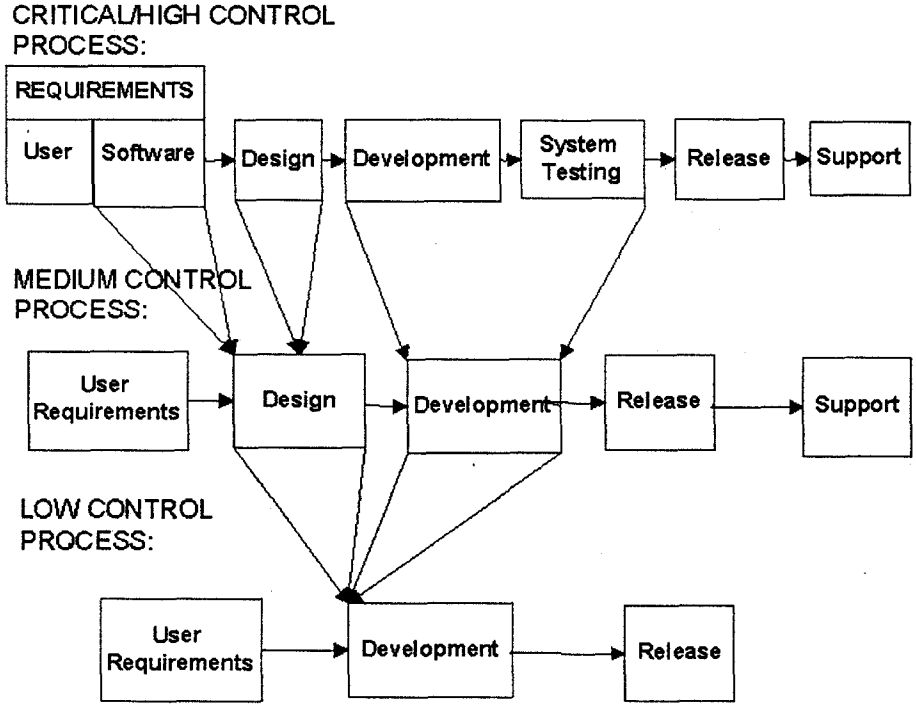

Figure 2, Control Level Phases

Determine Tailoring of Development Practices (ISO 9001, CMM Processes)

Organizations that have implemented ISO 9001 processes need to tailor the processes to the requirements of the project. Smart organizations have a

documented plan for tailoring their activities to fit the scope of a project just as they have a method for tailoring the documentation requirements of the project. Table 2 shows the tailoring of documentation to control level development used with the control level processes in Figure 2

Figure 2 illustrates the tailoring function based on the required control level. Large, complex, risky projects, requiring High or Critical Control are developed with a typical waterfall life cycle approach with requirements, design, development and testing phases. Projects under this control level would typically be characterized by selections from the right two columns of the table. Flight Software for the International Space Station power system or an application to be used across the aeronautics industry would be two good examples.

Medium control levels utilize a development life cycle that includes system testing as part of the Development phase and combines User and Software Requirements Analysis into one activity. The same functions are performed as for a High control project but the phases are not as rigorously defined and controlled and documentation requirements are slightly reduced.

Low control levels combine the Design and Development phases into one. Again the same functions are performed but even less rigorously defined and only minimum documentation requirements are imposed.

Table 2, Documentation Requirement by Control Level

Key: ▲ Required ❒ Optional

Software Documentation for:	Low	Medium	High	Critical
Management Plan	▲	▲	▲	▲
Development Activities Plan	▲	▲	▲	▲
Verification Plan	▲	▲	▲	▲
Validation Plan	▲	▲	▲	▲
Organizational and Technical Interface Descriptions	▲	▲	▲	▲
Acquisition Activities Plan			❒	❒
Training Development Plan			▲	▲
Assurance Plan		▲	▲	▲
Risk Management Plan		❒	▲	▲
Configuration Management Plan		▲	▲	▲
Delivery and Operational Transition Plan			▲	▲
Product Specification			▲	▲
Concept documentation			▲	▲
Requirements documentation	▲	▲	▲	▲
Design documentation	▲	▲	▲	▲
Version description		❒	▲	▲
User's guide	❒	❒	❒	❒
Operational Procedures Manual	❒	❒	❒	❒
Procedures				
Testing Procedures	▲	▲	▲	▲
Safety Assurance Procedures			❒	▲
Security and Privacy Procedures			❒	▲
Certification Procedures		❒	❒	▲

Utilize Results of Previous Steps

Utilizing the estimates and the risks to the project and the organization that have been identified provides a basis for tailoring the development processes that will be followed. Many of the questions answered during the estimation process were also asked again during the control level identification process. By integrating the two activities into one you assure consistency in the results when the two are combined. Use these results as the basis for tailoring the ISO 9001 processes. Be even more efficient by combining them into a single application which provides an initial development cost and schedule estimate, a compilation of processes that will reduce the risks to the project, organization and provide the best path to success, a development plan, a quality assurance plan, schedule and estimate and an IV&V assessment.

Additional decision making matrices can be added to this process, i.e. NASA NPG 2820 contains an appendix for containing the criteria for implementing Independent assessments or IV&V on a project. What you will find when incorporating additional decision matrices to the COCOMO/Control Level foundation described above, is that many of the factors in the new matrix have already been addressed. The IV&V matrix included 15 decisions but only required the addition of four new questions to the interview process and tweaking the existing reports to incorporate the new results.

Adjust Resources and Project

Having done all that, based on what you thought you knew about the project, did it agree with your initial assessment? Now that the data gathering is done, it's time to look at what happened when all the parts were combined. Is it an elephant, a horse or something out of mythology?

If you find that it's not what you initially expected, then the next step is to look at what can be changed to make it more familiar or achievable. If cost or time required to develop is the issue, adjust the COCOMO factors. If the risks are too high or the controls too stringent, adjust the Control Level factors. Too expensive and controlled? Adjust the factors that affect them both first. Once, you've done all that you rationally can, then decide is it something your organization wants to attempt, or should you pass on it. If you decide to continue, document the decisions that you made and be prepared to act on them.

For example, if your project estimate is over the budget allowed and you planned on using a programming team with little experience (6 months) with the intended platform, and the languages and tools being used (1 year). By using a more experienced team with one year experience with the platform and three years experience with the language and tools, the estimated cost for the project will drop from $317,000 to $265,000.

Pact Id	Pact Title	Percent Time	Percent Cost	Absolute Cost	Absolute Time	Low Pact	Medium Pact	High Pact	Critical Pact
P9	Requirements	0	0	$0.00	0	X	X	X	X
P10	Authorization to proceed	0	0	$0.00	0	X	X	X	X
P11	Identify design/coding standards	0	0	$0.00	0	X	X	X	X
P12	Maintain Software Development Folder	0	0	$0.00	0		X	X	X
P13	Software Assurance reviews Management Plan	0	0	$0.00	0		X	X	X
P14	Implement Problem report and corrective action system	0	0	$0.00	0		X	X	X
P15	Management Plan approval	0	0	$0.00	0	X	X	X	X
P16	Documented requirements	0	0	$0.00	0	X	X	X	X
P17	Peer review of requirements	0	0	$0.00	0		X	X	X
P18	Conduct formal inspection of requirements	0	0	$0.00	0				X
P19	Software Assurance reviews requirements	0	0	$0.00	0			X	X
P20	Requirements approval	0	0	$0.00	0	X	X	X	X
P21	Peer review of plans	0	0	$0.00	0			X	X
P22	Implement Formal configuration management	0	0	$0.00	0			X	X
P23	Conduct Product Assurance Audits	0	0	$0.00	0			X	X
P24	Conduct Formal Reviews	0	0	$0.00	0			X	X
P25	Document approval of requirements and formal review	0	0	$0.00	0			X	X
P26	Customer approval of certification procedures	0	0	$0.00	0				X
P27	Conduct analyses of criticality and safety	0	0	$0.00	0				X
P28	Plan and schedule IV&V activities	0	0	$0.00	0				X
P29	Identify method for verification of safety critical functions and requirements	0	0	$0.00	0				X

Table 3, Requirements Phase Risk Mitigations

Create Initial Mitigation Set

Most, if not all, development efforts share the same or similar risks, and luckily, many of the activities performed during development are forms of risk management or mitigation, i.e. requirements reviews, formal inspections, testing, etc.. Using a standard process that tailors your development activities you can easily identify an initial set of mitigation activities for the project as shown in Table 3.

The table includes a list of all possible mitigation activities that could be performed during the Requirements phase. It also identifies which of the activities are required based on the control levels. These activities, selected for the project's control level, then become the initial mitigation set. Associated costs for these activities can also be identified. This will assist when determining if additional activities should be performed based on the cost to implement an activity versus the cost of the risk occurring or the cost of implementing a less expensive activity.

Outcomes of the First Phase

As a result of this phase, you know

- What the project will cost
- How long the project will take
- What controls are required
- What documents need to be generated

- What activities need to be performed
- An initial set of risk mitigations
- Additional risk mitigations which might be selected. If their costs have been identified then you can compare what their cost to the project might be versus the risks they address.

Identify and Mitigate Risks

The second phase of the approach allows the user to scrutinize and tailor the initial risk mitigation plan developed by the first phase.

This second phase brings into play pre-assembled knowledge that relates the risk mitigations of the development plan to known software development risks. Several visualizations of this information enable to user to comprehend the interrelationships between risks and mitigations. Through these visualizations the user can better comprehend the risk landscape, and make a judicious choice of development activities. The main goal of this phase is to develop a risk mitigation plan that minimizes risk subject to the constraints on the software development effort (notably, schedule and cost).

Upon conclusion of this phase, the tailored development is transferred back to the Ask Pete tool for report generation.

In the subsections that follow we describe the main steps of this second phase, notably:

- Identify Risks
- Prioritize Risks
- Identify Risk Mitigations
- Estimate Risk Mitigation Effectiveness
- Select Risk Mitigations

Identify Risks

The tool is pre-populated with a detailed taxonomy of software development risks taken from the Software Engineering Institute (SEI). A snapshot of a fragment of this taxonomy is shown in, Figure 3:

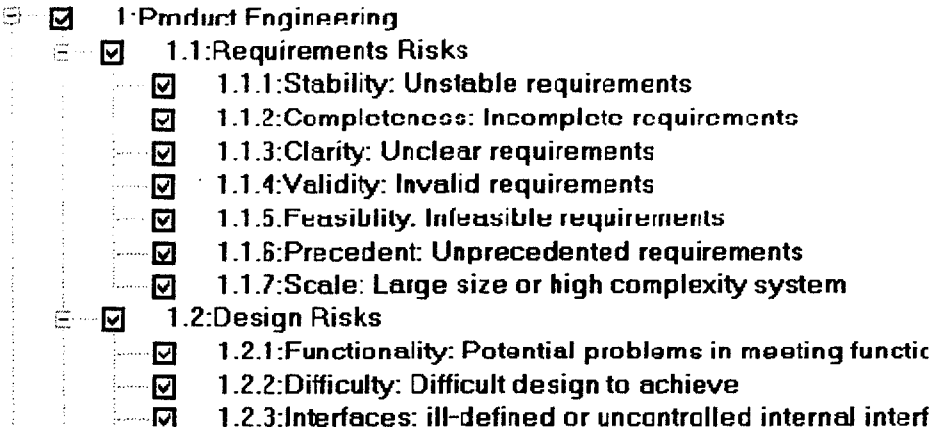

Figure 3, Risk Taxonomy

The user can tailor this risk taxonomy by discarding risks, and by adding new ones.

Discarding risks: For example, the user might discard a risk that is clearly irrelevant to the project at hand, but should be prepared to justify this action. The tool provides capabilities appropriate to support this practice, by offering a means to switch between views of retained and discarded risks, and by allowing the user to attach commentary to any risk (whether discarded or not). Thus a review team could quickly locate which risks had been discarded by the project team, and scrutinize their justifications for doing so.

Adding new risks: Users can add new risks, presumably risks not present in the SEI risk taxonomy. This enables them to extend the capabilities of the tool to operate on their specific concerns.

This hybrid approach of supplying pre-populated information, together with allowing for user customization, is a recurring theme of this phase. In this particular step of risk identification, the pre-populated risk taxonomy serves as a checklist to remind users of the broad spectrum of risks associated with software development, while the option to add new risks is a means by which users can add risks specific to their task/project/institution.

The net result of this step is a set of software development risks of concern to the task at hand.

Prioritize Risks

It is standard practice to evaluate risk as the combination of likelihood (probability of occurrence) and impact (how much damage it will do if it occurs). Our approach follows this practice – each individual risk has an a-priori likelihood figure (the probability of occurrence were nothing done to inhibit that risk), and separate rating of impact. For rating the impact, we offer a choice between a simple scheme in which the users directly assert their estimations of each risk's total impact, and a more elaborate scheme in which users relate risks to project requirements, and derive the impact from these relationships.

This more elaborate scheme requires further input from the users:

- Identifying the project requirements (which can be individually weighted to reflect their relative importance)
- Quantitatively relating the risks to the requirements that they impact – briefly, the impact of a risk on a requirement is a measure of how much of that requirement would be lost were that risk to occur.

Once these inputs have been provided, the tool automatically computes the total impact of a risk as the sum of the impacts on the weighted requirements.

Advantages of this more elaborate scheme derive from the creation of explicit traceability between risks and requirements: it serves as a disciplined way to make estimates of risk impact, and it results in an understanding of which requirements are most at risk. For example, it is possible to use this information to justify a renegotiation of the requirements themselves, in the case that some of the requirements are seen to be at risk, and mitigation of that risk is particularly expensive. Most importantly, this process allows multiple experts' knowledge to be combined to yield a consistent risk prioritization.

However, elaborating the requirements, and relating them to the extant risks, is a time-consuming process. In the cases where the users already have a good understanding of the relative impacts of the risks, they may prefer to follow the simper scheme and enter those values directly.

The net result of this step is a task-specific prioritization of the previously identified software development risks.

Identify Risk Mitigations

Recall that the first phase of the approach had yielded a set of suggested risk mitigation activities. The second phase presents to the user *all* the possible risk mitigation activities known to the first phase, indicating which of these have been suggested for this particular project. These mitigations, like the risks, are organized into a taxonomy for ease of scrutiny and navigation. In the figure below, suggested activities are indicated by the presence of check marks as shown in Figure 4.

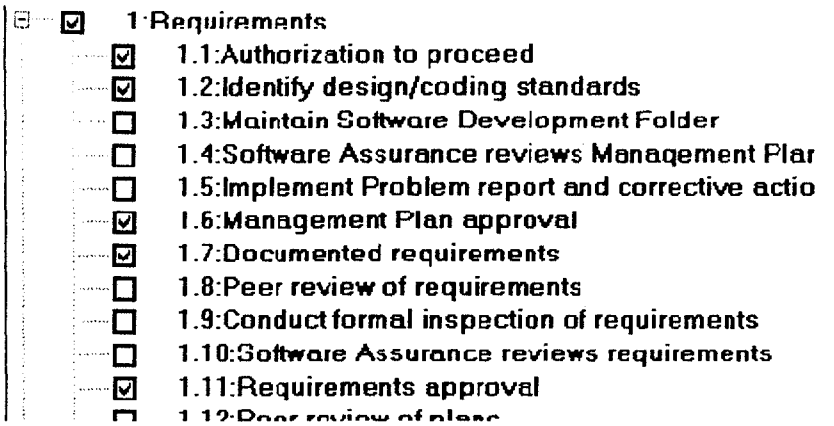

Figure 4, Mitigation List

The user can tailor this mitigation list by discarding mitigations, and adding brand new ones unknown to the first phase. Observe again the hybrid of pre-populated knowledge, in the form of a checklist of known activities, and the option for augmenting this with task-specific additions. Refining the selection from this universe of activities is to be addressed in the last step of this phase.

There are costs associated with performing mitigations. Most notably, budget and schedule are constrained resources in almost all software development efforts, and it is important to know how a suggested set of mitigations will consume these. The first phase of our approach yields cost and schedule estimates for each of the mitigation activities, and these estimates are carried over into this second phase. If users choose to add in additional mitigations, it is their responsibility to provide the costing information.

The net result of this step is a set of risk mitigation activities from which the users can choose.

Estimate Risk Mitigation Effectiveness

Typically, a mitigation will effect only a subset of all the risks, and of those, some more than others. Our approach accommodates this. Mitigations are cross-linked to the risks they mitigate, and associated with those links are quantitative estimates of the effectiveness of those mitigations.

The tool is pre-populated with quantitative effectiveness links between the Ask Pete universe of activities, and risks of the SEI software development risk taxonomy. We have made reasonable estimates of these effectiveness values, which we expect to refine as our experience base grows. Figure 5 shows a fragment of this cross-linking of activities to risks. Activities are the rows, and risks the columns. The numerical values in the white-background cells denote the effectiveness of the activity on the risk; this can range from 0.0 (not effective whatsoever) to 1.0 (fully effective at mitigating that risk). An empty cell is equivalent to an entry of 0.0

PACTxFM Col = Stability: Unstable requirements
Row = Authorization to proceed

		FMs	[-]Product Engineering							
		FMs	[-]Requirements Risks							[-]Design Ri
		FMs	Stabilit	Compl	Clarity	Validit	Feasib	Precec	Scale:	Functic	Diffi
PACTs	PACTs	FoM\R	0.2835	0.0405	0.0405	0.3645	0.2835	0.2205	0.2657	0.0255	0.021
	Authori	7.95	0.1	0.1	0.1	0.1	0.1	0.3	0.1		
	Identify	2.5								0.3	0.3
	Mainta	0									
	Softwa	2.65									
	Implem	1.85	0.9	0.3	0.9	0.9	0.3	0.3	0.1		
	Manag	0.15									
	Docum	1.7	0.3	0.9	0.9	0.1	0.3	0.3	0.1	0.1	
	Peer	2.85	0.9	0.9	0.9	0.9	0.9	0.9	0.1	0.1	
	Conduc	2.85	0.9	0.9	0.9	0.9	0.9	0.9	0.1	0.1	
	Softwa	2.8	0.9	0.9	0.9	0.9	0.9	0.9	0.1	0.1	
[-]Req	Requir	0									

Figure 5, Effectiveness Links

The users can adjust the effectiveness values, and annotate them with justifications as they do so. For example, the users might judge that formal inspections will be more effective than the pre-populated data would suggest, because they have an in-house team that is skilled in their application, and past experience has shown their high effectiveness in this area.

If the users have added new risks and/or new mitigation activities, then they will be required to cross-link those new items. For example, if the users add in a mitigation of using model checking (an effective analysis technique that has emerged from the formal methods research community over the last decade or so), they will have to assess which risks it mitigates (e.g., flaws in protocols), and how effective it is at doing so.

The net result of this step is a quantitative cross-linking of mitigations to the risks they address.

Select Mitigations

The primary purpose of this second phase is to allow the users to optimize the suggested program of activities suggested by Ask Pete. The steps up to this point have served to gather much of the key information upon which this optimization is based. However, optimization is not an automated step, rather, the users themselves are expected to explore the options and choose accordingly.

Several visualizations of this information enable to user to comprehend the interrelationships between risks and mitigations. Through these visualizations the user can better navigate the risk landscape, and be supported in making their judicious choice of development activities. A fragment of one of these visualizations follows.

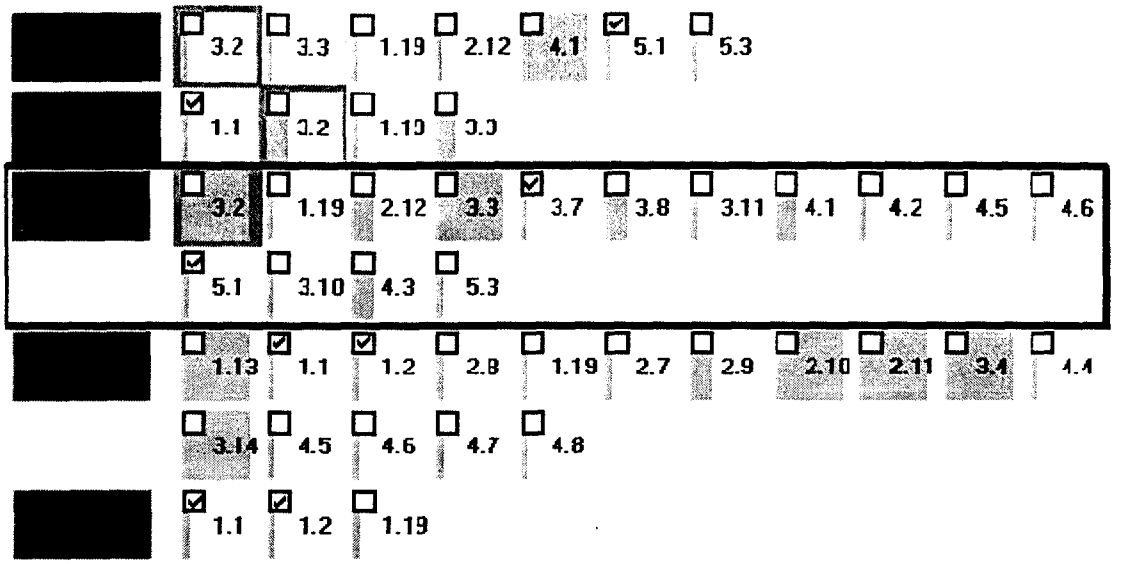

Figure 6, Risks and Mitigation Activities

Figure 6 shows graphically the risks, and for each risk, the activities available to mitigate that risk.

Each risk is displayed as one of the red-colored rectangles in the leftmost column, labeled with the risk number (e.g., 1.4.2) from the corresponding risk taxonomy. The width of the red bar is proportional to the logarithm of the risk. They have been automatically sorted into descending order, hence the width of the red bars diminishes towards the lower part of the figure.

The row to the right of each risk's rectangle displays the activities available to mitigate that risk, labeled with the mitigation number (e.g., 3.2) from the corresponding mitigation taxonomy. If there are too many mitigations to fit into one row, they spill over into a second row, and so on. Each activity has a small check box showing whether or not that activity has been selected, and through which the user can toggle that selection. The width of the turquoise rectangle is proportional to the effectiveness of that mitigation at reducing that risk, for example, mitigation 3.2 alongside risk 1.4.1 (the third row) has a relatively wide rectangle, indicating that it is highly effective at reducing that particular risk; in contrast, 3.2 alongside risk 1.4.2 (the top row) has a relatively narrow rectangle, indicating that it has only a small effect at reducing that risk.

The tool uses a variety of dynamic techniques to provide further detail, highlight selected items of focus (e.g., the border around risk 1.4.1 and its two rows of mitigations), etc.

In general, there is a non-trivial amount of information that users must take into account to make judicious selection of risk-mitigating activities. It does not appear to be feasible to display all this relevant information in one view. Instead, the tool's several visualizations offer a variety of forms of presentation, and a variety of means to focus in on different subsets of the information.

The net result of this step, and therefore of this entire second phase, is the selection of a set of activities together with the costing information associated with those activities. In the course of this second phase, the users could have adjusted the recommended selection that emerged from the first phase in several ways:

- Unselecting originally recommended activities
- Selecting additional activities from base set of mitigation activities
- Adding activities that were not options for the first phase to suggest
- Adjusting the cost and schedule estimates that were made by the first phase.

While the users are making their customizations during this second phase, they may record rationale for these actions. For example, if the users give a risk a relatively low priority, they may wish to record their justification for doing so. Similarly, if they alter the effectiveness and/or cost values of a risk mitigation activity, they again may wish to record a justification (e.g., assert that the cost would be higher than usual because they do not have any staff on hand already trained in that activity, and so would need to expend additional time and budget if they were to apply activity). Such user-supplied justifications are recorded as textual notes, and retained in the database. Users are not required to supply these, but are encouraged to do so both to serve as a reminder to themselves for future reference, and to serve as descriptive rationale to others (e.g., independent revivers of the project plan).

The information of the selected activities and adjusted cost and schedule estimates is transferred to the third phase for generation of various plans and reports, discussed next.

Combine and Implement Mitigations

Once the risk mitigation has been tailored in the second phase, the results replace the initial mitigations from the first phase. New mitigations are added and rejected mitigations are removed. The resulting costs of the mitigations in time and money are totaled and presented with the project estimate and control level information. The results are then incorporated in various plans, tailored to the particular project.

Publish Plans

Currently one report and two types of plans can be generated based on the results of this methodology, a Development Plan and a Product Assurance Plan. The report specifies the results of the planning and estimation activities, providing COCOMO II values, Control level, documentation requirements, IV&V recommendation and a narrative describing the development activities to be performed based on the control level. Each of the plans utilize these results by tailoring a detailed plan for a critical control project and removing irrelevant requirements and information. These plans should be further tailored by the project manager to add information relevant to the project and organization, i.e, personnel, unique processes, etc. The majority of the

tailoring can be performed in the templates used to generate the Product Assurance Plan.

Once this is completed, the plans should be made available to pertinent people and organizations.

Development Plan

The development plan, based on MIL-STD-498 and Data Item Description DI-IPSC-81427, addresses each of the development phases and documentation to be generated by the project.

Product Assurance Plan and Level of IV&V, Independent Assessment or IV&V

The Product Assurance Plan addresses typical Product assurance activities that should be performed during each of the phases and estimates the amount of time required to perform these tasks and evaluations. It also identifies the level of IV&V the project requires from independent assessment to full IV&V activities.

Conclusions

This approach combines all of the above to yield the outputs necessary to put together a successful project, namely:

- Identified software risks. The risk lists will take into consideration risks associated with software failures on previous NASA and aerospace missions (lessons learned, failure reports, defect profiles, etc.). This includes the identification of software components and intermediate deliverables by level of system criticality.
- An optimized plan that identifies software IV&V approach, development, and QA activities that mitigate and eliminate software risk for a given project at various times during the lifecycle.
- Consistent cost and schedule risk reduction budget estimates that establish a responsible balance between constrained project funding and the safe implementation of software subsystems.
- An equitably negotiated IV&V, Software Development, and QA plan that includes the priorities of the primary stakeholders while maintaining a high integrity program.
- IV&V, QA, and project plans that are compliant with institutional policies, ISO based Software Development Process Descriptions, and best practices from (for example) CMM.

The implementation of the tool support for the above process accommodates existing best practices for the various elements of project planning, estimation and assessment.

Acknowledgments

The research described in this paper was undertaken by Science Applications International Corporation (SAIC), and the Jet Propulsion Laboratory, California Institute of Technology, under contracts with the National Aeronautics and Space Administration, and by NASA Glenn Research Center. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government, the Jet Propulsion Laboratory, California Institute of Technology, NASA Glenn Research Center, or SAIC.

The authors gratefully acknowledge the contributions, guidance and assistance of Steve Cornford, Robert Dugas, Marcus Fisher, Michael Greenfield, Frank Huy, Hoh In. Jim Kiper, Tim Larson, Kenneth McGill, Tim Menzies, Burton Sigal, and Siamak Yassini with special appreciation to John Kelly and Martha Wetherholt for their support of the research.